Map generation: Map is generated in package data.ai by class MapGenerator and afterwards checked for validity by MapValidator class. The main decision made here was to restrict algorithm to create water fields on borders. There are 30 attempts to generate a valid map. When the map is valid, it is sent to server.

Algorithms: PathFinder uses Dijkstra algorithm to find the most optimal path to the Target tile. MinimalCoverageFinder uses greedy algorithm in order to find minimal coverage. Greedy algorithm allows us to find good enough coverage in adequate amount of time by making locally best decision in order to obtain globally acceptable result.

Network: ClientManager communicates with Network using ServerMessageConverter and ClientMessageConverter to convert the provided by LV classes to the local project ones.

Network class saves and instance of GameInfo class in order to communicate with Server.

ClientManager class saves instances of GameInfo and PlayerInfo classes to registerPlayer and send Map and Moves correctly.

MVC: Controller is implemented in package communication as ClientManager class and implements all the main logic and structure of the game.

CLI View is implemented in visalization package with CLI Class. CLI uses java.beans.PropertyChangeEvent and java.beans.PropertyChangeListener for event exchange with ClientManager and GameLogic.

Model is implemented with data packages. Data.map package contains classes connected with map, main classes are FullMap and MapTile.

GameLogic is a part of data.logic package, it implements the main business logic of the game. It uses State Maschine pattern for different Game Phases.

Error handling:

For Communication: Network error handling InvalidRequestException unchecked exception and for Converter PlayerNotFoundException unchecked exception.

For AI: NoSuchTileException and PathNotFoundException unchecked exceptions were implemented to handle errors cases specific for custom logic of application such as path finding and basic map validation before further logic will be applied.

MapGenerationException is checked exception used in case AI cannot generate correct map multiple times.

GameLogic uses InvalidGameState exception, that is unchecked exception used for show that game state is invalid.

Unit tests: 137 Unit Tests were created for all the classes, except Network and ClientManager, because they need a valid and actual gameID to perform correctly. Tests are written for both positive and negative cases, as well as using Mockito.